Running GEOSgcm on 480 cores using ...

swartzbr 12 posts since

Dec 11, 2007

As described in this ticket:

19827: failed attempts to run quarter degree GCM on 480 cores

The error message seen when attempting to run GEOSgcm using Intel MPI on 480 cores is:

error(0x30000): OpenIB-cma: could not register DAPL memory region for receive buffer: DAT_INSUFFICIENT_RESOURCES()

In

summary, the workaround to this problem is to alter the user's .cshrc

to set the stacksize limit to be unlimited, i.e. add this line to your

.cshrc:

limit stacksize unlimited

Here is the email trail for this issue:

Tom: This option

appears to be available in Intel 10 and not in Intel 9. GMAO still

uses Intel 9 for production runs. Of course we can test this option in

the Intel 10 compiler but the results may not be useful yet until GMAO

is ready to shift to that compiler.

-Hamid

Tom Clune wrote:

All,

Since this is not an OpenMP code, the most likely culprit fordriving up stack size in a Fortran application is large "Automatic"

arrays. Some compilers have a switch that changes the behavior of

automatic arrays that forces them to come from heap. Intel even has a

flag for this:

-heap-arrays <size>

I suggest that someone try recompiling with this flag with sizeset to a few sensible values and see what works (without using large

limit in the environment). Also should watch performance. Balance is

probably for some modest value of <size>.

- Tom

On Nov 2, 2008, at 7:48 PM, Oloso, Amidu O. (GSFC-610.3) AMTI wrote:

Nick,

There are two issues:

Issue 1: Only stacksize needs to be set to unlimited (very high value?)
for the code to run. All other limits remain the default set by the
system. On the harpertown compute nodes, the default limits are:

```
cputime unlimited
filesize unlimited
datasize 15388544 kbytes
stacksize 15388544 kbytes
coredumpsize 0 kbytes
memoryuse 3985408 kbytes
vmemoryuse 19704240 kbytes
descriptors 100000
memorylocked 8218560 kbytes
maxproc 139264
```

If stacksize is changed to unlimited, things work.

Issue 2: As you mentioned, setting the limit in the PBS script is not
enough because that setting stays only on the head node and does not
propagate to other nodes in the allocation. If the limit is set in the
".cshrc" file however, it seems to make it to all nodes in the
allocation.

-Hamid

Nicko Acks wrote:
Hamid,
Thanks for the update. Can you send the output of the current limit
output from the running job?

```
ulimit -a
core file size (blocks, -c) 0
data seg size (kbytes, -d) 15388540
file size (blocks, -f) unlimited
pending signals (-i) 139264
max locked memory (kbytes, -l) 8218558
max memory size (kbytes, -m) 3985408
open files (-n) 100000
pipe size (512 bytes, -p) 8
```

```
POSIX message queues (bytes, -q) 819200
stack size (kbytes, -s) 15388540
cpu time (seconds, -t) unlimited
max user processes (-u) 139264
virtual memory (kbytes, -v) 19704240
file locks (-x) unlimited
```

This is what my account shows as default limits but I have a default of bash (I don't think there are differences but it is another thing to look into).

Here is what I have for default using tcsh:

```
limit
cputime unlimited
filesize unlimited
datasize 15388540 kbytes
stacksize 15388540 kbytes
coredumpsize 0 kbytes
memoryuse 3985408 kbytes
vmemoryuse 19704240 kbytes
descriptors 100000
memorylocked 8218558 kbytes
maxproc 139264
```

I am pretty sure that Dan K. did attempt to set vmemoryuse to unlimited but not sure about the others. Hard limits are:

```
limit -h
cputime unlimited
filesize unlimited
datasize unlimited
stacksize unlimited
coredumpsize 0 kbytes
memoryuse 3985408 kbytes
vmemoryuse unlimited
descriptors 100000
memorylocked 8218558 kbytes
maxproc 139264
```

so the only three limits that can be set to unlimited are "datasize", "stacksize" and "vmemoryuse". Also, this limit if changed within the

job script it will only be set on the head node of a job (the other
nodes in the allocation do not get the limits passed to it from the job
script using Intel MPI, this is a known issue):

http://software.intel.com/en-us/articles/intel-mpi-library-for-linux-tips-and-tricks-faq#A7

The note about using limits.conf to set these limits is not exactly true
in our environment. We are setting limits in the sshd startup and
pbs_mom startup as the memory sizes are different based on the node type
across the entire cluster.

-Nick

On Sun, Nov 02, 2008 at 05:18:09PM -0500, Amidu Oloso wrote:

Ok. It all comes down to limits. I can consistently get a failed run and
a successful run. Bill has his limits set to unlimited except for
vmemoryuse that he sets to ~6GB.
So it looks like the problem is due to the default system limits. I will
find out exactly which of the limits is causing the problem. The key
thing is perhaps the "mystery" is solved.

-Hamid

Amidu Oloso wrote:

I did what Dan K. did i.e. replace my .cshrc with Bill's and I am now
running tests. I'll let you know what I find.

-Hamid

Nicko Acks wrote:

Folks,
We have resources at Intel waiting to attempt to reproduce the 480 core
GEOS-5 gcm problem that everyone but Bill Putman has been able to
reproduce so far. I also have my management asking for updates by early
Monday morning (because progress with Intel MPI and GEOS may effect some
spending within the next couple of months). If any progress has been
made in isolating the problem in the .cshrc that is at the root of the
DAPL problems that Dan (and others) have seen with this test case please
let me know before 9am Monday if possible.

thanks
Nick Tags: geos-5, geos5, geos, geosgcm, intel